Visual Ordnance Recognition System Fiscal 1998 Year-End Report: Algorithms and Hardware Design

JPL Internal Report D-16331

Prepared for:
Robotics Research Group
Air Force Research Laboratory
Tyndall Air Force Base
Panama City, Florida 32403-5323

Prepared by:
Clark F. Olson

Contributors:
Todd E. Litwin
Roberto Manduchi
Larry H. Matthies

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Mail Stop 107-102
Pasadena, CA, 91109-8001

October 1, 1998

Contents

1 Introduction **3**

2 Algorithm description **4**
2.1 Design considerations 6
2.2 Pre-processing 7
2.2.1 Stereo 7
2.2.2 Smoothing 8
2.3 Hypothesis detection 10
2.3.1 Fisher's Linear Discriminant 10
2.3.2 Neural network 11
2.3.3 Color cone 11
2.3.4 Connected components 12
2.4 Verification 13
2.4.1 Hypothesis resampling 13
2.4.2 Parallel edge detection 13
2.4.3 Gaussian filter 14
2.4.4 Height evaluation 14
2.4.5 Contrast evaluation 15
2.5 Evidential reasoning 15

3 Algorithm evaluation **15**
3.1 Detection performance 16
3.1.1 Hypothesis detection 16
3.1.2 Smoothing 16
3.1.3 Verification modules 17
3.2 Computation time 22
3.3 Recommendations 23

4 Hardware **23**

5 Fiscal Year 1999 **23**
5.1 Goals 23
5.2 Budget 25
5.2.1 Workforce 25
5.2.2 Hardware 25
5.2.3 Overall 26
5.3 Schedule 26

6 Concluding remarks **27**

A Components list **27**

1 Introduction

Considerable progress has been made at the Jet Propulsion Laboratory during Fiscal Year 1998 in the area of visual ordnance recognition. Our primary area of study has been in the area of algorithm design and evaluation. In addition, the design for a real-time system to perform ordnance recognition has been completed. This report describes the progress that has been made during this period, as well as our plans for 1999.

We have concentrated on detection of BLU-97 ordnance, which is in current usage in U.S. military test ranges (Fig. 1). The body of this type of ordnance is cylindrical and it is 20 centimeters long, with a 6 centimeter diameter. When new, the ordnance is bright yellow in color, but it is often weathered in practice on the test range.

Figure 1: Image of BLU-97 ordnance acquired at a Nellis Air Force Base test range .

The scenario for which this system has been designed is as follows. A unmanned vehicle traverses a test range at a speed of approximately 5 mph following a pre-determined route in order to fully cover the test range. As the vehicle is traversing, the ordnance recognition system continually examines the terrain in front of the vehicle looking for instances of BLU-97 ordnance. The ordnance recognition system provides a video signal of the terrain in front of the vehicle, visually marking the locations of the detected ordnance. When ordnance is detected, the vehicle will stop (and retreat to a safe stand-off distance, if necessary), in order to perform remediation.

We have considered a large suite of algorithms to perform the ordnance recognition in this scenario, which have been divided into four categories: pre-processing, hypothesis detection, verification modules, and evidential reasoning. Figure 2 shows the flow of information between algorithms in these categories, as well as the algorithms in each category that have been examined. The images are first pre-processed in order to yield better detection performance. Candidate ordnance locations are then identified using one or more hypothesis detection methods. Verification modules are applied to the candidate locations in order to lower the rate of false positives. Finally, evidential reasoning techniques are used to combine

the information for the hypothesis detection and verification modules in order to make a final decision on each candidate.

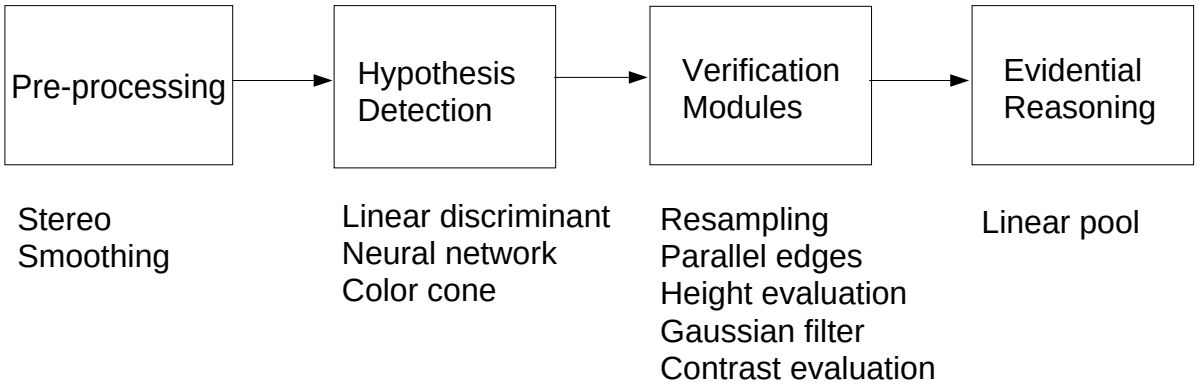


Figure 2: The overall algorithmic design. The boxes represent the type of algorithm applied. The listings below are instances of the type of algorithm.

For each algorithm that has been considered, we evaluate the techniques on a test set from Nellis Air Force Base [8] with respect to both the detection performance and the computation time required on a workstation. Using these evaluations, we make recommendations as to the suite of algorithms that should be included in the final system. The current algorithm design includes stereo preprocessing, hypothesis detection using a color cone discriminator, hypothesis resampling followed by several verification modules, and probabilistic evidential reasoning to combine the results of the verification modules.

Taking into account the suite of algorithms above we have completed a hardware design that will allow real-time operation (0.5-2 frames per second) of the method. This design includes a pair of progressive-scan, 3-CCD color cameras for input to a high-end PC/104 Plus pentium (or compatible) card running LynxOS. Output will be provided through a video signal (NTSC, S-video, and VGA output will be available) that displays the imagery from one of the cameras overlaid with highlights demarking the candidate ordnance locations. Serial communications will provide additional information with respect to the detections.

Following the discussion of our progress in 1998, we describe our plans for fiscal 1999 including schedule, milestones, and expected cost. Highlights of the plan include two data collections (one early in the year using digital cameras on a tripod, one later in the year using the real-time system) and field tests on both a vehicle at JPL and the AFRL vehicle. Our 1999 efforts will culminate with the delivery of a complete system in September.

2 Algorithm description

In this section we describe the suite of algorithms that has been considered for performing real-time detection of surface-lying BLU-97 ordnance. The algorithms have been divided into four categories:

1. **Pre-processing**
   Prior to the detection of candidate ordnance locations in the imagery, it is often useful to perform some pre-processing on the imagery in order to improve the quality of the data with respect to the detection performance, or to extract data that is useful in the detection process. Examples that we consider are (1) smoothing of the image in order to reduce noise and undesirable image texture, and (2) stereo triangulation of imagery from a pair of cameras in order to determine the range to the various locations in the image. Either, both, or neither of these could be selected for the the real-time system depending on the relative performance enhancement and computation time required.

2. **Hypothesis detection**
   We must have some method to select candidate locations in the image that could contain instances of the ordnance. For hypothesis detection, we have concentrated on methods that examine the color at the various image pixels, since this technique is fast and yields a low rate of false negatives in our test set. Each pixel is classified as either ordnance-like or non-ordnance-like according to its color. Candidates are located by detecting large sets of connected ordnance-like pixels. Several techniques have been considered to determine how to classify the individual pixels, including Fisher's linear discriminant, neural networks, and a convex color cone discriminator. A single hypothesis detection method should be selected. While more than one of these techniques could be used in combination, this would increase the processing time significantly, while detecting largely the same candidates.

3. **Verification**
   Once candidates positions have been located, further processing can be performed in order to reduce the rate of false positives (while largely retaining the true positives). At the candidate locations, we can use methods that require increased computational resources, since we examine only a small portion of the image during verification. Methods that have been considered for performing verification include parallel segment detection, correlation-based filtering, height evaluation, and contrast evaluation. An additional technique that is useful for the above verification methods is a resampling of the image at the candidate location such that the dominant orientation is rotated to a canonical orientation and range data is used to scale the ordnance to a canonical scale. Multiple verification techniques can be used since these are applied only to the candidate ordnance locations in the image.

4. **Evidential reasoning**
   Finally, the results of the verification modules (in the form of verification scores) must be combined together with the initial score from the hypothesis detection stage using some evidential reasoning process in order to make a decision on whether the candidate location should be reported to the operator. We have concentrated on a probabilistic combination model where each process generates an estimated probability together with a confidence value. The probabilities are combined according to the confidence values in order to generate the final score.

The next subsection describes some important design considerations for the overall system. The following subsections then discuss the algorithms considered in further detail.

2.1 Design considerations

In designing a real-time ordnance recognition system, it is important to consider the imaging geometry in order to ensure that the system has sufficient opportunity to detect the ordnance. Some important considerations are the swath-width, minimum and maximum range of detection, vehicle speed, and the number of pixels on target. We examine these issues here.

The height of the cameras on the vehicle is estimated to be 2 meters, and we assume that the ground plane is approximately flat for this analysis. Figure 3 shows a diagram of the imaging geometry. Given that we would like to be able to sense approximately in the range between $d_1 = 10$ meters and $d_2 = 30$ meters in front of the vehicle, and that we require a swath-width of approximately $l_1 = 3$ meters to ensure clearance of the vehicle, the field-of-view (FOV) should be as follows to yield the minimum swath-width at the closest range (10 meters):

$$\Theta = 2\sin^{-1}\frac{l_1/2}{\sqrt{d_1^2 + (l_1/2)^2}} = .2978 = 17.06^\circ \tag{1}$$

For the cameras to observe the furthest range (30 meters), this implies that the camera tilt α should satisfy:

$$\tan\frac{\pi}{2} - \alpha + \frac{\Theta}{2} = \frac{d_2}{h} \tag{2}$$

$$\alpha = .2155 = 12.34^\circ \tag{3}$$

With these parameters, the closest visible range will be:

$$d_0 = h\tan\frac{\pi}{2} - \alpha - \frac{\Theta}{2} = 5.24 \text{ meters} \tag{4}$$

The cross-range distance at the furthest distance is:

$$l_2 = 2d_2\sin\frac{\Theta}{2} = 8.90 \text{ meters} \tag{5}$$

A maximum vehicle speed of 5 MPH, coupled with a processing speed of no worse than 1 frame per 2 seconds implies that the vehicle will travel no more than 14.67 feet (4.47 meters) per iteration of the algorithm, which will give the system multiple chances to detect the ordnance while it is in the visible range.

The dimensions of the image will be 512×480. If the orientation of the ordnance is perpendicular to the cameras, then for an instance of BLU-79 at a range of 10 meters, the number of pixels on target in the horizontal direction will be:

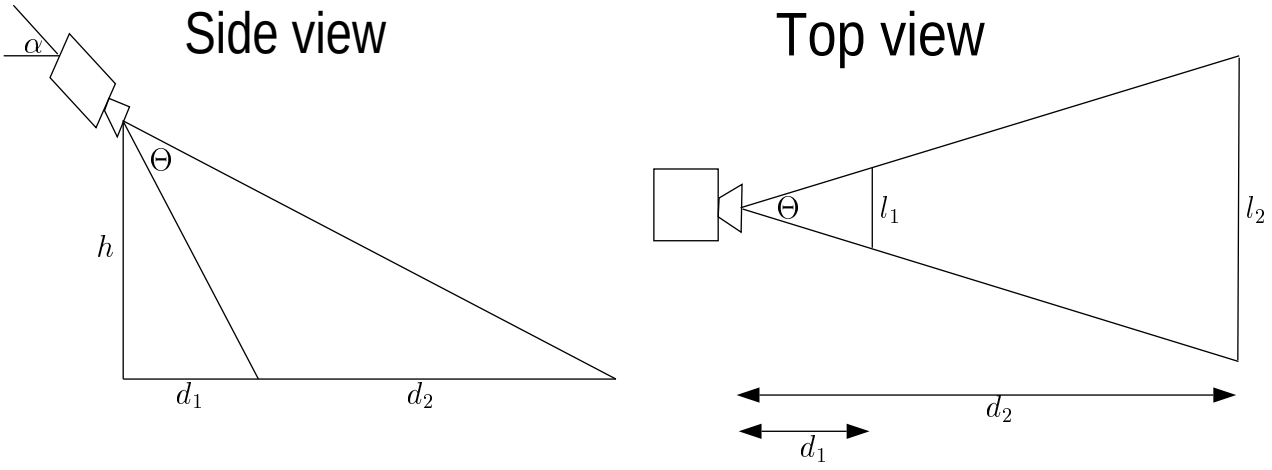


Figure 3: Schematic of the camera geometry. The side view shows the feasible range of detection and the top view shows the swath width at the various ranges. There are three degrees of freedom in this system (e.g. α, θ, and h).

$$p_{h_1} = \frac{w}{l_1} \times 512 \text{ pixels} = 34.1 \text{ pixels} \tag{6}$$

and the number of vertical pixels on target will be:

$$p_{v_1} = \frac{2r}{l_1} \times 512 \text{ pixels} = 10.2 \text{ pixels} \tag{7}$$

At the furthest range, these dimensions become:

$$p_{h_2} = \frac{w}{l_2} \times 512 \text{ pixels} = 11.5 \text{ pixels} \tag{8}$$

$$p_{v_2} = \frac{2r}{l_2} \times 512 \text{ pixels} = 3.5 \text{ pixels} \tag{9}$$

2.2 Pre-processing

The techniques that we consider for pre-processing the image data prior to hypothesis detection are smoothing and stereoscopy. The following subsections describe these methods.

2.2.1 Stereo

The use of stereo range data [6, 7] is anticipated to be crucial to the ordnance recognition system that will be developed. This information can be used for several purposes:

1. Bounds can be placed on the size of the ordnance at any location in the image, thus reducing both the search space and the number of false positives found in the hypothesis detection stage.

2. A hypothesis resampling step can be performed that transforms each hypothesis detected into a canonical frame of reference such that each hypothesis has the same scale.

3. The appropriate scale can be selected at each location in the image for smoothing and edge detection, if desired.

4. Obstacle detection can be performed using the range data, if desired.

In order to perform stereo range mapping, an off-line step, where the stereo camera rig is calibrated must first be performed. We use a camera model that allows arbitrary affine transformation of the image plane [12] and that has been extended to include radial lens distortion [3]. The remainder of the method is performed on-line.

At run-time, each image is first warped to remove the lens distortion and the images are rectified so that the corresponding scan-lines yield corresponding epipolar lines in the image. The disparity between the left and right images is measured for each pixel by minimizing the sum-of-squared-difference (SSD) measure of a window around the pixel in the Laplacian of the image. Subpixel disparity estimates are computed using parabolic interpolation on the SSD values neighboring the minimum. Outliers are removed through consistency checking and smoothing is performed over a 3×3 window to reduce noise. Finally, the coordinates of each pixel are computed using triangulation.

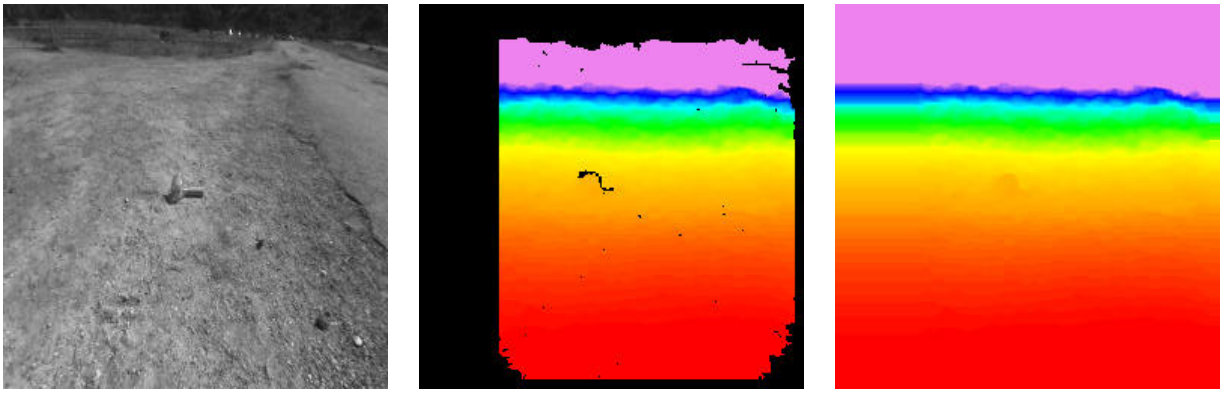
Note that not every pixel is assigned a range with this method. There are a variety of factors that result in some pixels not being assigned a range including occlusion, window effects, finite disparity limits, and outliers. Despite this problem, we desire a range estimate at each pixel in the image. To resolve this problem, we propagate the range values from neighboring pixels using a simple method that prefers neighbors to the left or right to those above or below.

Figure 4 shows an example of the range data computed using these techniques.

2.2.2 Smoothing

A basic smoothing operation can be used to reduce noise in the image and to blur undesirable image textures by convolving the image with a Gaussian filter with some specified scale σ. However, this ignores the fact that the image phenomena occur at widely varying scales due to perspective effects in the image. An alternative is to perform variable-scale smoothing guided by range data [9].

Our approach to performing adaptive smoothing is to filter the image at multiple scales and then interpolate the response for each pixel at the appropriate scale given by the range data. We must first select an appropriate set of scales to use. Based on the size of the ordnance and the ranges over which we desire accurate recognition in the test imagery, we have chosen to work with scales in the range $0.8 \leq \sigma \leq 3.2$, and we perform smoothing at three scales ($\sigma_1 = 0.8, \sigma_2 = 1.6, \sigma_3 = 3.2$) in order to interpolate any scale in the range accurately. At each scale, we filter with the Gaussian derivative in both x and y. Each image is thus convolved with 6 filters.



(a) (b) (c)

Figure 4: Range data extracted from a stereo pair. (a) Left image of a stereo pair. (b) Distance from the camera mapped into color values. Red is closest to the camera and magenta is furthest. Black pixels indicate no valid range data. (c) Range map after filling unknown values with estimates.

Now, we must interpolate the response at each pixel for the appropriate scale given by the stereo range information. Since there is an inverse linear relationship between the range to an object and its scale in the image, we map the range information into a scale for smoothing and edge detection using:

$$\sigma(x, y) = K/R(x, y), \tag{10}$$

where $R(x, y)$ is the range estimate given by the stereo processing at the pixel, and K is a constant determined by the size of the object of interest (the ordnance in this case).

We approximate the correct response at each pixel using parabolic interpolation (separately for x and y) in the $\ln\sigma$ domain. Let $F_\sigma(x, y)$ be the desired response at (x, y) for the scale σ. In determining the equation that yields the appropriate response, it is useful to perform a coordinate transform such that $z = \log_2 \frac{\sigma(x,y)}{\sigma_2}$. For $\sigma_1 = \frac{1}{2}\sigma_2 = \frac{1}{4}\sigma_3$, this yields $z_1 = -1$, $z_2 = 0$, and $z_3 = 1$. With this transformation it is simple to show that:

$$\begin{aligned}
F_\sigma(x, y) &\approx az^2 + bz + c && (11)\\
a &= \frac{1}{2}(F_{\sigma_1}(x, y) - 2F_{\sigma_2}(x, y) + F_{\sigma_3}(x, y)) && (12)\\
b &= \frac{1}{2}(F_{\sigma_3}(x, y) - F_{\sigma_1}(x, y)) && (13)\\
c &= F_{\sigma_2}(x, y) && (14)\\
z &= \log_2 \frac{\sigma(x, y)}{\sigma_k} && (15)
\end{aligned}$$

2.3 Hypothesis detection

Several methods have been consider for detecting candidate ordnance locations using color processing. The methods described in the following subsections examine the color at each pixel in the image and classify it as either ordnance-like or non-ordnance-like. The ordnance-like pixels are grouped using a connected components techniques and hypotheses are identified by finding large connected components of ordnance-like pixels.

2.3.1 Fisher's Linear Discriminant

We have implemented an adaptive version of color-based recognition based on Fisher's linear discriminant. The color statistics of the ordnance are learned off-line, while those of the background are estimated at run-time. This allows the discriminator function to vary with the image properties. The disadvantage to this adaptive approach is that the computation time is greater, since the discriminator function can't be pre-compiled into a lookup table.

The basic idea of Fisher's linear discriminant (see, for example, [2]) is to project the data vectors onto a line by taking the dot product of each with a carefully chosen vector such that the two populations have the maximum separation possible. A point on the line can then be chosen to be the boundary between the decision classes. If the vector projects to one side of the point it is classified as ordnance and if it projects to the other it is classified as background. This is the same as dividing the vector space with a plane and using this plane to discriminate between the two populations. If the plane is chosen correctly, good classification results can sometimes be achieved.

Fisher's linear discriminant can be formulated as follows. We have samples from two distributions, the ordnance pixels and the background pixels. Call the set of ordnance samples O and the set of background pixels B. Each individual sample is a vector of the three color coordinates that make up the pixel. Let m_o and m_b be the means of the sample populations. The within-class scatter matrices are defined by:

$$S_o = \sum_{x \in O} (x - m_o)(x - m_o)^t \tag{16}$$

$$S_b = \sum_{x \in B} (x - m_b)(x - m_b)^t \tag{17}$$

and

$$S_w = S_o + S_b \tag{18}$$

According to Fisher's linear discriminant, the carefully chosen vector that we should use for the dot-product is given by:

$$w = S_w^{-1}(m_o - m_b) \tag{19}$$

2.3.2 Neural network

A neural network can be used to perform the classification between the ordnance-like and non-ordnance-like pixels. A subset of the imagery has been classified manually and used for training such a classification method using the back-propagation algorithm. We have used a neural network simulator called NevProp3 [4] to train the network. We chose to use a network with 3 input units (the basic color coordinates), 5 hidden units, and 1 output unit specifying the likelihood that the pixel is ordnance-like (Fig. 5).

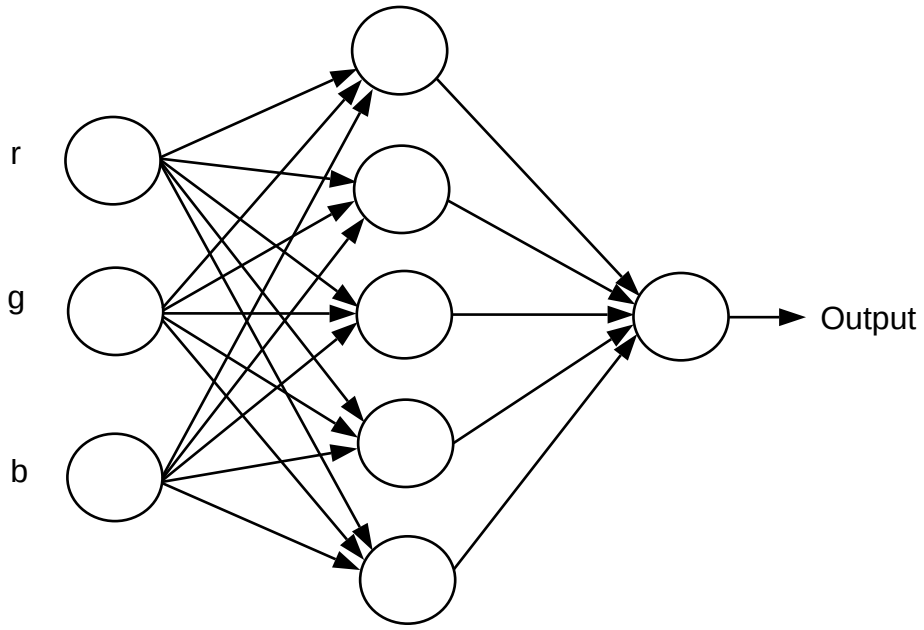


Figure 5: Network structure for classifying pixels.

In practice, the output value of the neural network for the various inputs are compiled into a look-up table to allow fast processing at run-time. A disadvantage to the neural network method is that the learned values are non-intuitive to understand and cannot be easily tuned by hand under changing conditions.

2.3.3 Color cone

Another method that we can use to classify pixels is to define a convex polyhedron in the color space and classify those pixels that fall into the polyhedron as ordnance-like and those outside as non-ordnance-like. This subsection develops one such method of doing so using a polyhedral cone in the color space.

It has been shown that normalized color-space coordinates are independent of the scene geometry [5]. This means that for scene points illuminated by the same spectral power distribution, the orientation of the scene point, the orientation of the illumination, and the

overall brightness of the illumination do not affect the color coordinates. This property holds for any color coordinate that is the ratio of two of the basic color coordinates (r, g, and b). Using such ratios will thus yield invariance to scene geometry and illumination brightness, if we assume that the spectral power distribution of the illumination in the scenes in which we are interested is constant.

We can define a cone in the color space by specifying an upper and lower bound on the ratio of each pair of basic color coordinates:

$$t_{rb} \geq \frac{r}{b} \geq T_{rb} \tag{20}$$

$$t_{rg} \geq \frac{r}{g} \geq T_{rg} \tag{21}$$

$$t_{gb} \geq \frac{g}{b} \geq T_{gb} \tag{22}$$

The cone in the color space yielded by these equation has a convex cross-section consisting of six sides (one for each constraint).

Now, we must determine the upper and lower bounds on these ratios that will yield good classification of the ordnance-like pixels. This can be performed through training using examples. We have used a simple method that approximates gradient descent search in order to bring the rate of false positive and false negatives into an equilibrium where the rates are approximately the same.

In addition, we impose a constraint on the overall brightness of each pixel that has the effect of truncating the cone:

$$r^2 + g^2 + b^2 < t_{rgb} \tag{23}$$

This rules out areas of the image that are too dark to yield accurate classification results.

As with the previous method, we can compile the resulting classification function into a look-up table for fast processing at run-time.

2.3.4 Connected components

After classifying each of the pixels according to one of the above criteria, we detect the connected regions of the image that have been classified as ordnance-like according to the color. Detecting the connected components can be performed in linear time in the size of the image using a version of the union-find data structure [1]. This method uses a two-pass algorithm to detect the large, connected regions of ordnance-like pixels in the image. If the size of the region is larger than some threshold T (which is a function of the range to the location), then the location is considered to be a candidate for further verification.

2.4 Verification

After detecting candidate ordnance locations, a variety of verification modules can be applied to the candidates in order to reduce the likelihood of detecting a false positive instance in the imagery. We can apply much more computation in these verification modules than in the initial hypothesis generation step since we have greatly reduced the area of the image in which we are interested. This subsection describes several such verification techniques that we have examined.

2.4.1 Hypothesis resampling

A first step that is useful prior to applying various verification modules is to compute the dominant orientation in the image at the location of the hypothesis and to resample the image using this information and the stereo data such that the resampled image is at a canonical scale and orientation.

We detect the dominant orientation in the image by applying a simple gradient operator that examines each pixel and three of its' neighbors and then histogramming the gradient orientations found (weighted by the gradient magnitude) at each pixel. The histogram bin with the largest score is taken to be the dominant orientation of the hypothesis.

For each pixel in the resampled image, we then compute the corresponding location in the original image according to:

$$x = \frac{s_d}{s_h}(x_i - x_h)\cos\theta + \frac{s_d}{s_h}(y_i - y_h)\sin\theta \tag{24}$$

and

$$y = -\frac{s_d}{s_h}(x_i - x_h)\sin\theta + \frac{s_d}{s_h}(y_i - y_h)\cos\theta, \tag{25}$$

where θ is the dominant orientation, s_d is the desired scale, s_h is the scale of the hypothesis according to the range data, x_i and y_i and the coordinates in the resampled image, and x_h and y_h and are the position of the hypothesis in the original image. Since, the computed x and y are continuous values, we determine the pixel values using bilinear interpolation:

$$x_0 = \lfloor x \rfloor \qquad x_1 = x_0 + 1 \qquad \alpha = x - x_0 \tag{26}$$

$$y_0 = \lfloor y \rfloor \qquad y_1 = y_0 + 1 \qquad \gamma = y - y_0 \tag{27}$$

$$I(x,y) = (1-\alpha)(1-\gamma)I(x_0,y_0) + (1-\alpha)\gamma I(x_0,y_1) + \alpha(1-\gamma)I(x_1,y_0) + \alpha\gamma I(x_1,y_1) \tag{28}$$

2.4.2 Parallel edge detection

An initial attempt at detecting ordnance found the candidates by locating pairs of parallel segments in the image edge map [10]. While it was found that this technique was too slow and too error-prone to be used as a hypothesis detection stage, a simple version can still be used as one of several verification modules.

The first step in this technique is to detect the gradients in the resampled candidate window. The gradients are then histogrammed according to their orientation. If an orientation is found where the score is very high, this indicates that either a single, very strong, straight edge is present, or a pair of strong parallel edges are present. This information is used to evaluation the hypotheses.

2.4.3 Gaussian filter

Since the candidate windows have been resampled to a canonical scale and orientation, a simple measure of the similarity between the window and a piece of ordnance is a matched filter, which in this case would be a yellow rectangle placed on a background of the average color terrain. However, since the position, orientation, and scale of the window are estimates, and, in addition, foreshortening will cause the length of the rectangle to be unknown, we have instead used a filter consisting of Gaussian derivatives. This allows for errors in the estimation of these variables and yields increased robustness. Our filter consists the product of a Gaussian second derivative in the y-direction (across the cross-section of the ordnance) with a Gaussian in the x-direction (across the length of the ordnance). The filter is thus given by:

$$F(x,y) = \left(\frac{y^2 - \sigma_y^2}{2\pi\sigma_x\sigma_y^5}\right) e^{-\frac{x^2}{2\sigma_x^2} - \frac{y^2}{2\sigma_y^2}} \tag{29}$$

and its appearance can be seen in Figure 6.

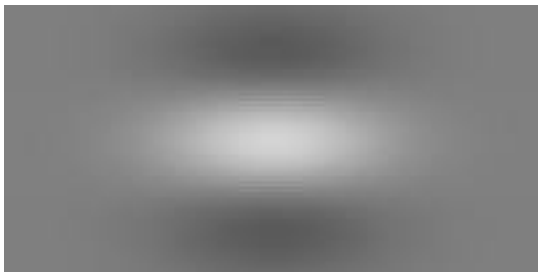


Figure 6: Gaussian derivative filter.

The filter is applied only to the red band of the color image, since this band contains most of the useful information for discriminating ordnance from background.

2.4.4 Height evaluation

When we have stereo range data, it is sometimes possible to detect the difference in height of the terrain at the location of an ordnance instance. A simple technique that we use is to examine the range data corresponding to the pixels in the candidate windows and determine

the minimum and maximum heights present. The difference in these values yields a measure of the amount of height variation in the window.

2.4.5 Contrast evaluation

A final indicator that we use to help verify instances of ordnance is the contrast present in the candidate window, which we measure by the maximum gradient present, since we expect the true instances of the ordnance to yield a significant gradient between the ordnance and the background. While false positives might also yield such a high gradient, this will not always be the case, and we can thus use this information to help discriminate between true positive and false positives.

2.5 Evidential reasoning

Once the various verification modules have generated scores for each of the candidate ordnance locations, we must have some method for combining the scores into a single measure that can be used to evaluate each candidate. We use a linear opinion pool (see, for example, [11]), where the results of each measurement are combined according to a weighting factor that is determined along with the probability result. The weighting factor represents the confidence in the probability estimate that is generated.

Let H be the hypothesis that a certain candidate actually represents an ordnance instance. Each verification module v yields a probability value $P_v(H)$ that the hypothesis is correct and a weighting factor $W_v(H)$. We can combine the values from any two verification modules (for example v_1 and v_2) using the following relationships:

$$P_{v_1+v_2}(H) = \frac{W_{v_1}(H)P_{v_1}(H) + W_{v_2}(H)P_{v_2}(H)}{W_{v_1}(H) + W_{v_2}(H)} \tag{30}$$

$$W_{v_1+v_2}(H) = W_{v_1}(H) + W_{v_2}(H) \tag{31}$$

Since these equations are associative, it does not matter in which order the values are combined; the final result will be the same. The candidate is finally accepted if, after combining all of the scores from the various verification modules, the probability value above a pre-determined threshold.

3 Algorithm evaluation

This section examines the performance of the algorithms described above with respect to both the detection performance and the computation time required. Recommendations are made as to the algorithms to use in the final system.

3.1 Detection performance

We first examine the performance of the algorithms with respect to the ordnance detection rate versus the false alarm rate. This performance is captured by receiver-operating characteristic (ROC) curves that plot the detection rate on one axis versus the false alarm rate on the other as the detection threshold is varied. Such curves allow the threshold to be chosen such that the tradeoff between the detection rate and the false alarm rate is optimized. The ideal ROC curve rises very quickly until the detection rate is nearly unity with few false positives and then flattens as false positives become more likely with a decreasing threshold.

3.1.1 Hypothesis detection

Among the algorithms, the hypothesis detection stage is perhaps the most important and we thus evaluate it first. We assume for this evaluation that stereo is used, since it is crucial to setting the detection threshold appropriately (instances at farther ranges are allowed to have lower a threshold on the number of ordnance-like pixels, since they appear smaller in the image). However, we do not use any smoothing, yet. The effects of smoothing are evaluated subsequently.

Training set In order to evaluate the hypothesis detection methods, we have created a training set of ordnance images from a set of imagery collected at a Nellis Air Force Base test range [8]. The training set consists of 10 images that have been pre-selected. For each image, the regions corresponding of ordnance instances were found manually by noting the corners of a quadrilateral that fit the instance well. Each pixel inside one of the quadrilaterals was considered to be part of the ordnance instance. Any pixel not inside any of the quadrilateral was considered to be part of the background. It should be noted that the quadrilaterals were roughly rectangular in order to fit the body of the ordnance. In general, the small chute was sometimes still attached to the ordnance was considered to be part of the background for classification purposes. Figure 7 shows two example images.

Results The results obtained by the hypothesis detection methods can be seen in Figure 8. The color cone method for detecting the hypotheses is clearly superior to the other methods and the linear discriminant has the lowest performance. This is probably because the linear discriminant uses only a single plane in the color space to attempt to separate the classes. The color cone uses the intersection of seven planes in the color space to define a narrow region and is formulated in such a way that the illumination effects are automatically negated. The neural network can approximate these effects, but not easily.

3.1.2 Smoothing

We now examine the effect that smoothing has on the performance of the hypothesis detection stage. The color cone method of hypothesis detection is used for this test. Two types of smoothing are considered. First, a conventional, Gaussian smoothing operator with $\sigma = 2$

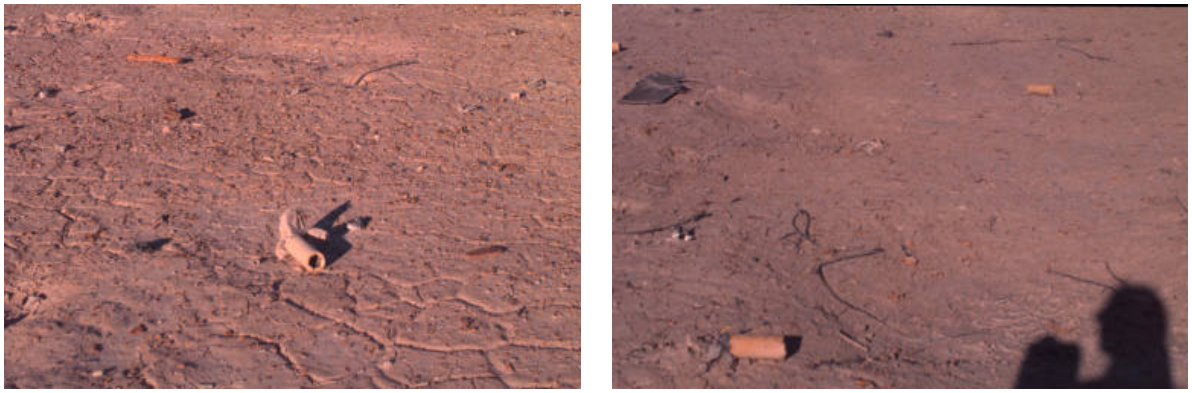


Figure 7: Example images from the data set collected at Nellis Air Force Base.

pixels was applied to the test imagery (including the training images). Second, we used the variable-scale smoothing guided by stereoscopy described above. Figure 9 shows the ROC curve for these two cases, as well as the case with no smoothing. It can be observed that the smoothing does not significantly improve the hypothesis detection results.

3.1.3 Verification modules

Four verification modules have been tested with respect to the performance improvement that is gained over using solely the hypothesis detection stage. These verification modules were described in the previous section:

- Gaussian filter
- Parallel edge detection
- Height evaluation
- Contrast evaluation

The use of the Gaussian filter verification module and the parallel edge detection technique also require the use of the hypothesis resampling technique. The other techniques do not require hypothesis resampling.

Figure 10 shows the results of applying each of the verification modules separately. It can be seen that the use of each of the verification techniques improves the performance of the detector, since the curves with verification mostly lie above the curves with no verification. Figure 11 shows the results of using all of the verification modules together. The results are certainly better than not using any of the verification modules, but it is not completely clear that all of the verification modules are necessary, since the overall result is only slightly superior to some of the individual verification modules. Further analysis on the data set

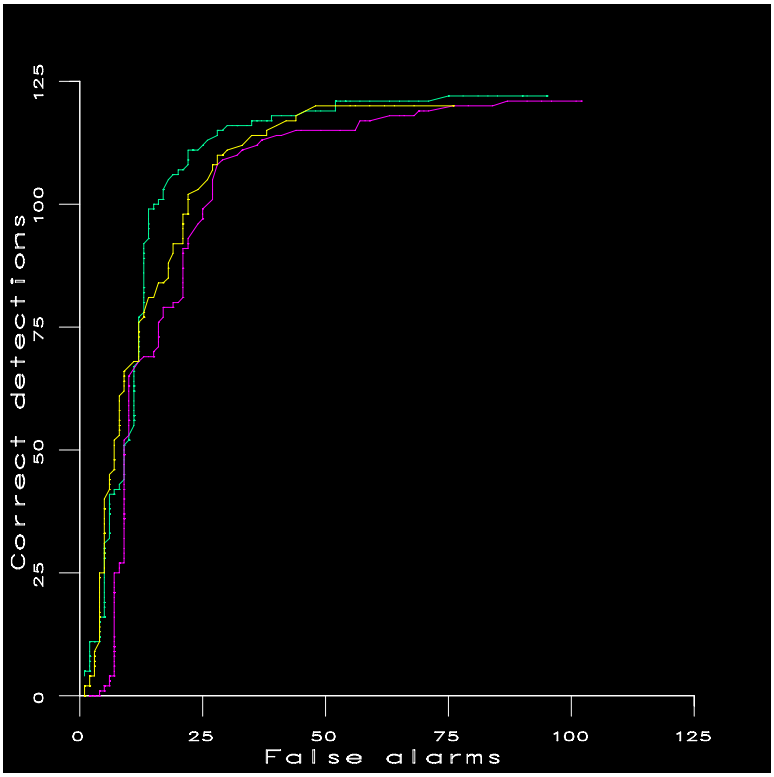


Figure 8: Receiver-operating characteristic curve showing the performance of the hypothesis detection techniques. The green curve shows the performance of the color cone technique. The purple curve shows the performance of the neural network method. The yellow curve shows the performance of the linear discriminant function.

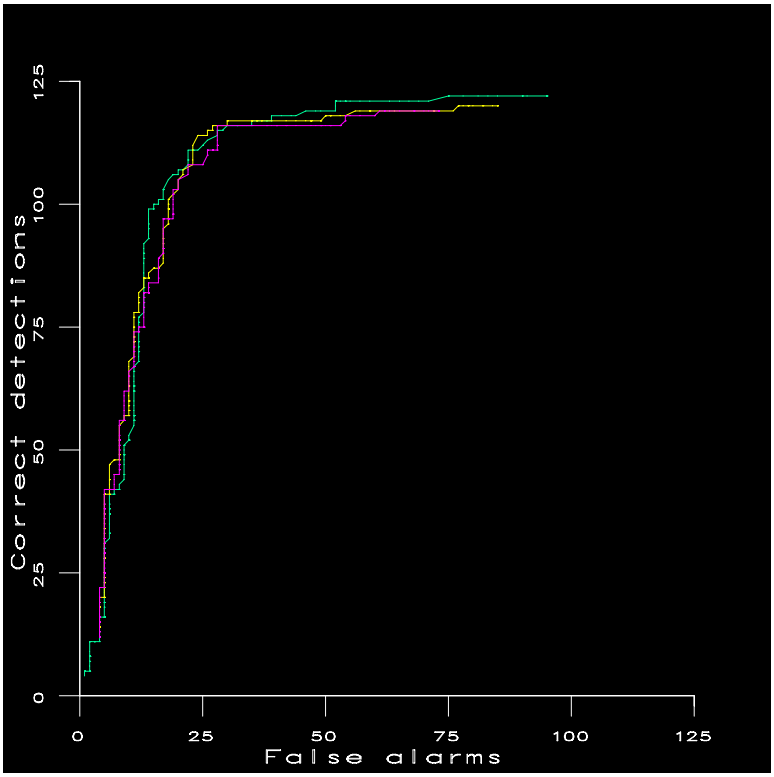


Figure 9: Receiver-operating characteristic curve showing the performance of the color cone technique with various types of smoothing. The green curve is the case with no smoothing. The yellow curve is conventional smoothing. The purple curve is for the case with adaptive smoothing according to the range data.

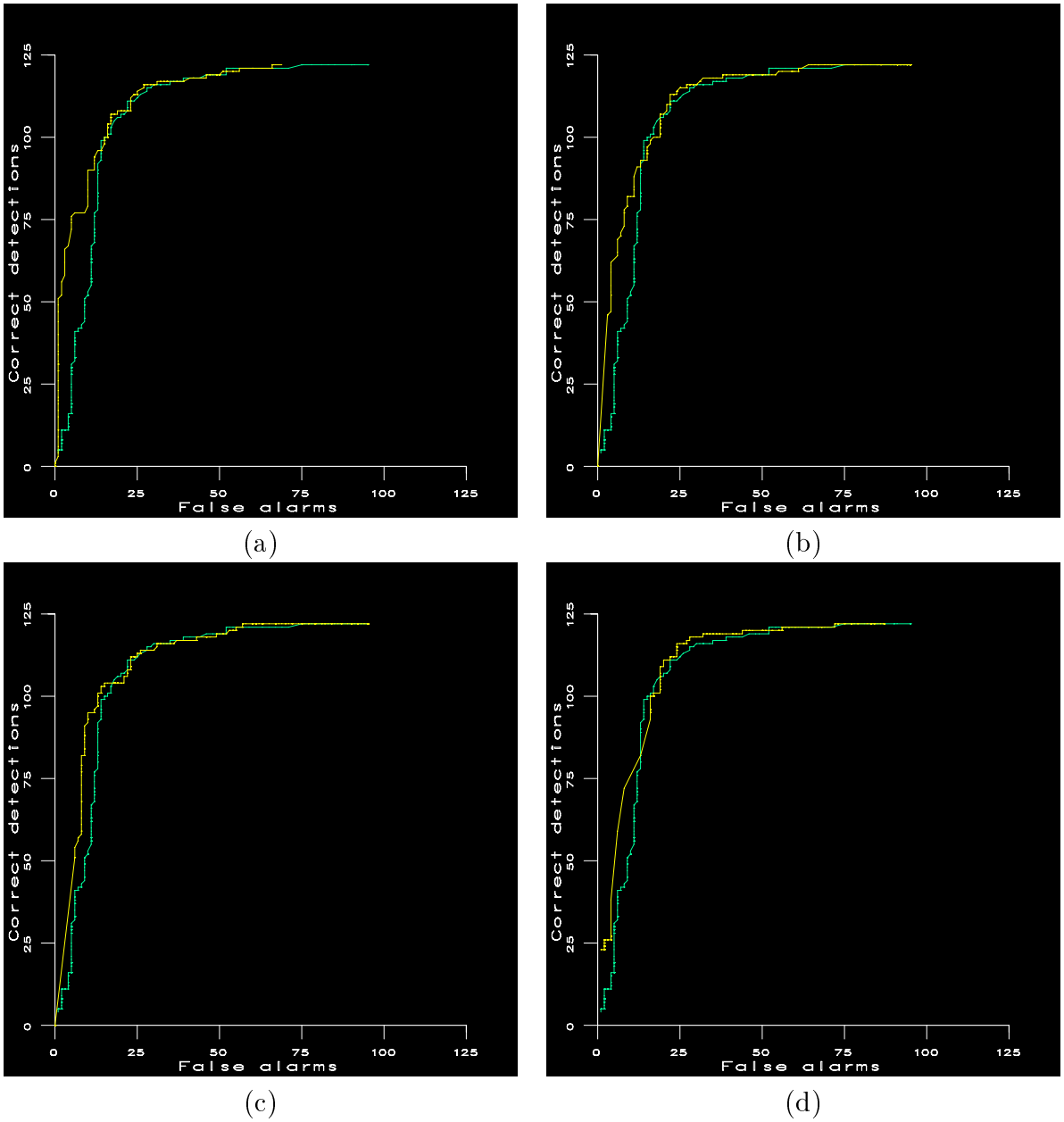


Figure 10: Receiver-operating characteristic curves showing the performance of the verification modules combined with the color cone techniques for hypothesis detection. In each case the green curve is the case with no verification modules and the yellow curve is the case using a single verification module. (a) Gaussian filter (b) Parallel edge detection (c) Height evaluation (d) Contrast evaluation

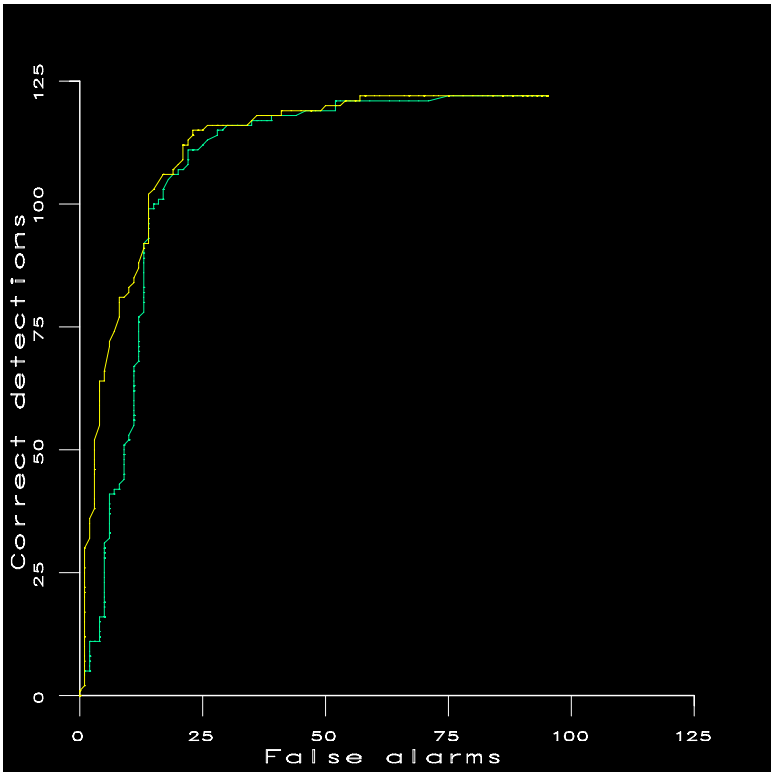


Figure 11: Receiver-operating characteristic curves showing the performance of of all of the verification modules. The green curve is the case with no verification modules. The yellow curve is the case with all of the verification modules.

to be collected early this fiscal year will allow us to determine what combination of these modules provides the best cost/performance ratio.

3.2 Computation time

In addition to the detection performance, we must strongly consider the computation time required by the algorithms, since running in near real-time (0.5 to 2 frames per second) is necessary to accommodate a vehicle speed of of 5 MPH. Table 1 shows the performance that has been observed on a SPARCstation™20 for the algorithms we have considered. We note that the SPARCstation™20 has a 60 MHz clock speed and we expect the techniques to run considerably faster on the 233 MHz pentium compatible chip that we have selected for use in the real-time system.

Module	Algorithm	Time required
Pre-processing	Stereo	0.900† seconds per image
	Smoothing	0.800† seconds per image
	Adaptive smoothing	2.500† seconds per image
Hypothesis detection	Neural network	0.624 seconds per image
	Color cone	0.624 seconds per image
	Linear discriminant	1.720 second per image
Verification	Hypothesis resampling	0.103 seconds per hypothesis
	Gaussian filter	0.008 seconds per hypothesis
	Contrast evaluation	0.045 seconds per hypothesis
	Height evaluation	0.003 second per hypothesis
	Parallel edge detection	0.075 second per hypothesis
Evidential reasoning	Linear opinion pool	0.001 second per image

† Estimate.

Table 1: Processing time required by various algorithm modules as measured on a SPARCstation™20.

The stereo pre-processing technique requires significant computation, but the results of the stereo pre-processing are important to the detection capabilities and are indispensable if obstacle detection is desired. The smoothing techniques also require signification computation, and since they do not appear to yield a significant improvement in the detection performance, they should not be included in the final algorithm.

The hypothesis detection methods that can be compiled into a look-up table (the neural network and the color cone techniques) require the same computation time to detect candidate locations, since they use the same code, once the look-up table has been computed. The linear discriminant method requires more time, since it computes the discriminator function on-line.

The verification modules all require relatively little computation time per hypothesis that has been located. We thus recommend the use of all of them until they can be evaluated further on a larger test set, such as the one we will be collecting early this fiscal year. In addition, the evidential reasoning method is necessary, but requires very little computation time.

3.3 Recommendations

Based on the detection and computational performance described above, it is our recommendation that algorithms used on the final system include stereo but not smoothing for pre-processing, the use of the color cone hypothesis detection method, and all four of the verification modules, in addition to the linear opinion pool for evidential reasoning.

4 Hardware

We have completed a preliminary design of the hardware necessary to process the data at a rate of 0.5-2 frames per second (Fig. 12). This design includes two color cameras and frame-grabbers to input images for stereo processing. The CPU we have selected is a 233 MHz Pentium PC/104 Plus board and we anticipate using the Lynx operating system. The CPU will perform lens control through RS-232 communications with the cameras.

Two PC/104 PC card drives will be used. One for storage of the operating system and other necessary system files and the other to store images for data collection. A power converter will be used to provide power to the PC/104 stack from the 12V external input and a cooling fan is necessary to prevent overheating.

The CPU will output results through VGA graphics and RS-232 communications. A scan converter will convert the VGA graphics to NTSC video (or optionally S-video) output. We note that this design includes processing of the data up until the generation of the video signal displaying possible ordnance locations and the serial output of additional information. This design does not handle the communication of this data between the vehicle and the operator control station or the display of the information at the operator control station.

A list of the components that we have identified for use on the real-time system appears in the appendix.

5 Fiscal Year 1999

This section discusses our goals, schedule and budget for fiscal year 1999.

5.1 Goals

In the upcoming year, we expect to achieve the following goals:

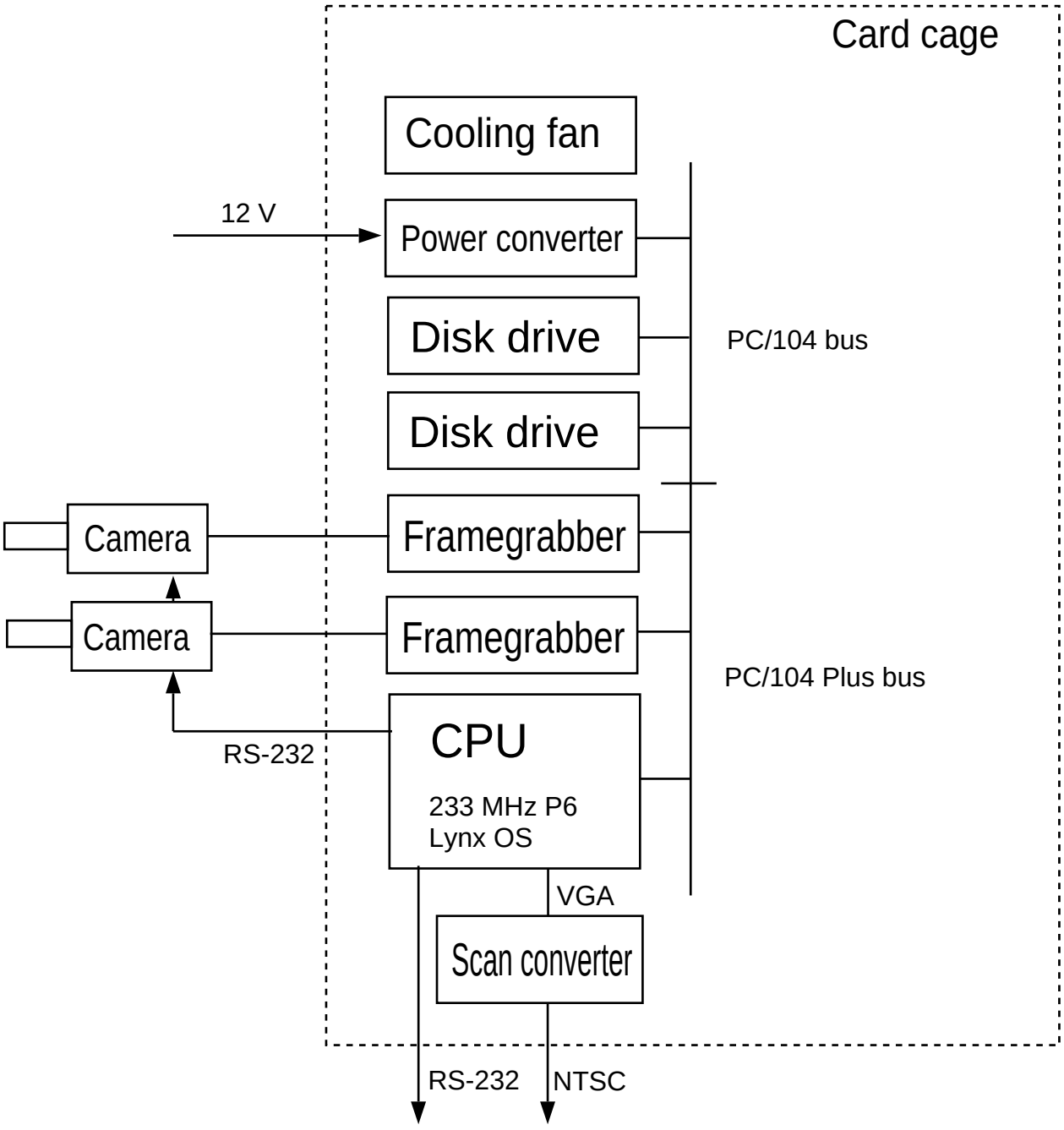


Figure 12: Design of the real-time system.

- Conduct another, more extensive data collection at Nellis that mimics the clearing operation as it would actually be performed, with sequences of images taken along straight lines at short intervals. This allows us to both validate the performance of the algorithm and better characterize the performance of the method with respect to the range to the target. This data collection would use the color stereo pair of digital cameras the we purchased last year.
- Assemble the hardware system that will go on the vehicle (including interfacing between cameras, frame-grabbers, CPU and other cards, mounting cameras on bar, calibrating cameras, assembling cards and fan in card cage, and testing video output and disk drive).
- Port code to real-time system. We will use C to implement the software for the real-time system.
- Tune parameters and test the system stand-alone at Nellis AFB. The nature of the color-based front-end implies that using a new set of cameras (and frame-grabbers) with the lenses covered by a colored filter will require a different look-up table to identify potential ordnance locations. This final data collection will also validate the performance of the complete architecture.
- Test the system on a vehicle at JPL. This test will be as close as we can make it to the nominal clearing scenario. We will mount the system at the height and orientation expected for the final system and drive at the expected speed of the AFRL vehicle toward a BLU-97 mockup at a variety of locations and orientations.
- Conduct a field test with the hardware on the AFRL vehicle.

5.2 Budget

We have compiled a tentative budget for fiscal year 1999.

5.2.1 Workforce

Our workforce levels are expected to be:

Clark Olson	0.50 work-years
Roberto Manduchi	0.25 work-years
Systems person	0.25 work-years
Jim Lloyd	0.02 work-years
Estimated cost:	\$180,000

5.2.2 Hardware

The combined hardware cost of the system outlined above is approximately \$40,000.

5.2.3 Overall

We estimate that an additional $20,000 should be sufficient to cover our remaining expenses including travel, reserve, equipment and miscellaneous expenses such as computer services, supplies, telephone charges, duplication services, and freight.

Our overall budget for the year is thus expected to be:

Workforce	$180,000
Hardware	$ 40,000
Travel/Reserve/Misc	$ 20,000
Total:	$240,000

5.3 Schedule

We expect our schedule for fiscal 1999 to be roughly as follows:

- October, 1998:

  - Finish second pass at design of hardware system, based on final results of algorithm performance evaluation and runtime evaluation.

  - Finish year-end report.

  - Travel to Panama City to present and discuss current results.

- November, 1998:

  - Collect data set that mimics clearing operation

  - Finalize hardware design and make procurements.

- December, 1998:

  - Finish evaluation of algorithms on new data set.

- January, 1998:

  - Complete training in LynxOS.

- February, 1999:

  - Complete and test interface between between CPU, cameras, frame-grabbers, and video output.

- March, 1999:

  - Finish porting algorithms to selected CPU.

- April, 1999:

  - Data collection using real-time system at Nellis AFB.

- May, 1999:
  - Run-time benchmarking of complete system.
  - Finish evaluation of system on April data collection.
- June, 1999:
  - Test on JPL vehicle using ordnance mock-up.
- July, 1999:
  - Conduct field test with hardware on AFRL vehicle. (Date negotiable.)
- September, 1999:
  - Deliver complete system.

6 Concluding remarks

Considerable work has been accomplished in 1998 towards the generation of the real-time system to perform ordnance recognition. We have identified algorithms that are successful in detecting instance of BLU-97 ordnance with a low rate of false positives and characterized the performance of these algorithms with respect to the detection rate on a dataset from Nellis Air Force Base and with respect to the computation time on a workstation. This performance analysis has led to the generation of a set of recommendations for algorithms to be included on the real-time system. A preliminary design for the system has been completed that will allow the algorithms to run at the desired rate in order be used on a moving vehicle. These accomplishments have laid the foundations for the completion of the real-time system for ordnance recognition in 1999.

Acknowledgements

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government, or the Jet Propulsion Laboratory, California Institute of Technology.

A Components list

The components that we believe are necessary for developing the real-time system are as follows:

Component	Brand	Part(s)	Cost
CPU	Ampro	233 MHZ CPU Accessories	

Description: Single-board computer with 233 MHZ Pentium CPU and PC/104 Plus expansion site, and DRAM.
See: http://www.ampro.com/products/littlebd/lbp5x.htm

Component	Brand	Part(s)	Cost
Operating system	Lynx	Open Development Env.	$6995
		12 months support	$3500
		Licenses	$3500

See: http:/www.lynx.com/products/lynxos.html

Component	Brand	Part(s)	Cost
Cameras	Sony	3-chip color camera	2 × $5444
		Power supply	2 × $ 154
		Power cables	2 × $ 52
		Video out cables	2 × $ 94
		Remote control unit	$ 611
		Lenses	2 × $1844

See: http://bpgprod.sel.sony.com/model.bpg?model=DXC9000

Component	Brand	Part(s)	Cost
Frame-grabbers	Imagenation	Color frame-grabber	2 × $ 595

See: http://www.imagenation.com/products/PXC200.pdf

Component	Brand	Part(s)	Cost
Video output	Aitech International	MultiPro Plus	$ 209

Description: External VGA-to-NTSC converter with 9v input.
See: http://www.aitech.com/multipro-new.htm

Component	Brand	Part(s)	Cost
Keyboard	Wireless Computing	Wireless surfboard	$ 399

Description: Radio-linked wireless keyboard and touchpad/mouse with 50 ft range. See: http://www.wireless-computing.com/

Component	Brand	Part(s)	Cost
Disk drive	Adtron	PC/104 drive	2 × $ 190

See: http://www.syspac.com/ãdtron/sddpover.htm

Component	Brand	Part(s)	Cost
Disk cartridge	Calluna Technology	520 MB PCMCIA card	2 × $ 423

Component	Brand	Part(s)	Cost
D/A converter	Diamond Systems	D/A board	$ 525
		Terminal block	$ 69
		Ribbon cable	$ 10

See: http://www.diamondsys.com/rmm1612.htm

Component	Brand	Part(s)	Cost
Flack shielding	McMaster-Carr	12x12x1/8 Polycarbonate	$ 10
		12x12x1/4 Polycarbonate	$ 15

Description: Cameras will be mounted behind sheets of Lexan. We need pieces to place over lenses for testing.

Component	Brand	Part(s)	Cost
Card cage	JPL Stores	Electrical box	
		Toggle switch	
		Face plate	
		LED	
		Mounting clip	

Component	Brand	Part(s)	Cost
Cooling fan	Nidec	TA225DC cooling fan	$ 8

Component	Brand	Part(s)	Cost
Power converter	Diamond Systems	DC/DC power supply	

See http://www.diamondsys.com/he104.html.

Component	Brand	Part(s)	Cost
Stereo mount	SLIK	Twin Plate 400	$ 213

Description: Padded bar to mount cameras on.

References

[1] M. B. Dillencourt, H. Samet, and M. Tamminen. A general approach to connected-component labeling for arbitrary image representations. *Journal of the ACM*, 39(2):253–280, April 1992.

[2] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[3] D. B. Gennery. Least-squares camera calibration including lens distortion and automatic editing of calibration points. In A. Grün and T. S. Huang, editors, *Calibration and Orientation of Cameras in Computer Vision*. Springer-Verlag, in press.

[4] P. H. Goodman. NevProp software, version 3. University of Nevada, Reno, 1996. URL: ftp://ftp.scs.unr.edu/pub/cbmr/nevpropdir/.

[5] G. Healey. Segmenting images using normalized color. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(1):64–73, January/February 1992.

[6] L. Matthies. Stereo vision for planetary rovers: Stochastic modeling to near real-time implementation. *International Journal of Computer Vision*, 8(1):71–91, July 1992.

[7] L. Matthies, A. Kelly, T. Litwin, and G. Tharp. Obstacle detection for unmanned ground vehicles: A progress report. In *Proceedings of the International Symposium on Robotics Research*, pages 475–486, 1996.

[8] C. F. Olson. A color stereo image set for evaluation of ordnance recognition algorithms. Technical Report Internal Report D-15655, Jet Propulsion Laboratory, California Institute of Technology, 1998.

[9] C. F. Olson. Variable-scale smoothing and edge detection guided by stereoscopy. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 80–85, 1998.

[10] C. F. Olson and L. H. Matthies. Visual ordnance recognition for clearing test ranges. In *Detection and Remediation Technologies for Mines and Mine-Like Targets III, Proc. SPIE*, 1998.

[11] F. Voorbraak. Combining evidence under partial ignorance. In *Proceedings of the First International Joint Conference on Qualitative and Quantitative Practical Reasoning*, pages 574–588, 1997.

[12] Y. Yakimovsky and R. Cunningham. A system for extracting three-dimensional measurements from a stereo pair of TV cameras. *Computer Vision, Graphics, and Image Processing*, 7:195–210, 1978.